

Terms used

hash near/4 table and **table** or **bank near/8 concurrent** or **simultaneous** or **parallel near/4 access** or **read** or **w**

Sort results by

Display results
 [Save results to a Binder](#)

Try a

 [Search Tips](#)

Try t

☐ Open results in a new window

Results 1 - 20 of 200


Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

1 [Programming languages for distributed computing systems](#)

Henri E. Bal, Jennifer G. Steiner, Andrew S. Tanenbaum

September 1989 **ACM Computing Surveys (CSUR)**, Volume 21 Issue 3

Full text available:  [pdf\(6.50 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [ci](#)

When distributed systems first appeared, they were programmed in traditional sequential language procedures for sending and receiving messages. As distributed applications became more common, the approach became less satisfactory. Researchers all over the world began designing new programming languages for distributed applications. These languages and their history, their underlying pr ...

2 [Simultaneous reference allocation in code generation for dual data memory bank ASIPs](#)

Ashok Sudarsanam, Sharad Malik

April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, v

Full text available:  [pdf\(156.30 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [ci](#)


We address the problem of code generation for DSP systems on a chip. In such systems, the amount of application software must be sufficiently dense. Additionally, the software must be written so as to meet constraints which may include hard real-time constraints. Unfortunately, current compiler technology is unable to handle architectures which are highly irregular. Thus, designers often r ...

Keywords: code generation, code optimization, graph labelling, memory bank assignment, register allocation

3 [Compiler transformations for high-performance computing](#)

David F. Bacon, Susan L. Graham, Oliver J. Sharp

December 1994 **ACM Computing Surveys (CSUR)**, Volume 26 Issue 4

Full text available:  [pdf\(6.32 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [ci](#)


In the last three decades a large number of compiler transformations for optimizing programs have been developed. Uniprocessors reduce the number of instructions executed by the program using transformations based on basic block flow techniques. In contrast, optimizations for high-performance superscalar, vector, and parallel architectures rely on tracking the properties of basic blocks with transformations that rely on tracking the properties of ...

Keywords: compilation, dependence analysis, locality, multiprocessors, optimization, parallelism, register allocation

4 [Design tradeoffs for the Alpha EV8 conditional branch predictor](#)

André Seznec, Stephen Felix, Venkata Krishnan, Yiannakis Sazeides

May 2002 **ACM SIGARCH Computer Architecture News**, Volume 30 Issue 2

Full text available:  [pdf\(1.24 MB\)](#)  [Publisher Site](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [ci](#)

This paper presents the Alpha EV8 conditional branch predictor. The Alpha EV8 microprocessor project, in development, envisioned an aggressive 8-wide issue out-of-order superscalar microarchitecture featuring a large number of registers and a large number of instructions in flight. The Alpha EV8 microprocessor project is a joint effort between Digital Equipment Corporation and Compaq Computer Corporation. The Alpha EV8 microprocessor is a 64-bit, out-of-order, superscalar processor. It is designed to support a wide range of applications, from embedded systems to high-performance servers. The Alpha EV8 microprocessor is a 64-bit, out-of-order, superscalar processor. It is designed to support a wide range of applications, from embedded systems to high-performance servers. The Alpha EV8 microprocessor is a 64-bit, out-of-order, superscalar processor. It is designed to support a wide range of applications, from embedded systems to high-performance servers.

multithreading. Performance of such a processor is highly dependent on the accuracy of its branch area was devoted to branch prediction on EV8. The Alpha EV8 branch pre ...



Keywords: EV8 processor, Branch Prediction

5 The effect of instruction fetch bandwidth on value prediction

Freddy Gabbay, Avi Mendelson

April 1998

ACM SIGARCH Computer Architecture News , Proceedings of the 25th annual conference on computer architecture, Volume 26 Issue 3

Full text available:  [pdf\(1.32 MB\)](#)  [Publisher Site](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citation](#)

Value prediction attempts to eliminate true-data dependencies by dynamically predicting the outcome of data-dependent instructions based on that prediction. In this paper we attempt to understand the limits of value prediction. We show that the instruction-fetch bandwidth and the issue rate have a very significant impact on the study of how recent techniques to improve the instruction issue rate can be applied.

6 Data and memory optimization techniques for embedded systems

P. R. Panda, F. Catthoor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkarni, A. Vandercappelle, P.

April 2001

ACM Transactions on Design Automation of Electronic Systems (TODAES), Volume 6 Issue 1

Full text available:  [pdf\(339.91 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citation](#)

We present a survey of the state-of-the-art techniques used in performing data and memory-related optimizations. These optimizations are targeted directly or indirectly at the memory subsystem, and impact one or more of the performance, power dissipation, and area of the resulting implementation. We first examine architecture transformations. We next cover a broad spectrum of optimization techniques.

Keywords: DRAM, SRAM, address generation, allocation, architecture exploration, code transformation, synthesis, memory architecture customization, memory power dissipation, register file, size estimation

7 Converting thread-level parallelism to instruction-level parallelism via simultaneous multithreading

Jack L. Lo, Joel S. Emer, Henry M. Levy, Rebecca L. Stamm, Dean M. Tullsen, S. J. Eggers

August 1997

ACM Transactions on Computer Systems (TOCS), Volume 15 Issue 3

Full text available:  [pdf\(526.39 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citation](#)

To achieve high performance, contemporary computer systems rely on two forms of parallelism: instruction-level parallelism (ILP) and thread-level parallelism (TLP). Wide-issue super-scalar processors exploit ILP by executing multiple instructions in parallel. Multiprocessors (MP) exploit TLP by executing different threads in parallel on different processors. Simultaneous multithreading (SMT) statically partitions processor resources, thus preventing thread contention.

Keywords: cache interference, instruction-level parallelism, multiprocessors, multithreading, simultaneous multithreading

8 A general framework for prefetch scheduling in linked data structures and its application to networked data structures

Seungryul Choi, Nicholas Kohout, Sumit Pamnani, Dongkeun Kim, Donald Yeung

May 2004

ACM Transactions on Computer Systems (TOCS), Volume 22 Issue 2

Full text available:  [pdf\(2.45 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citation](#)

Pointer-chasing applications tend to traverse composite data structures consisting of multiple independent pointer chains. A single pointer chain leads to the serialization of memory operations, the traversal of independent pointer chains in parallel. This article investigates exploiting such *interchain memory parallelism* for the purpose of *multi-chain prefetching*. Previous work has focused on *single-chain prefetching*.

Keywords: Data prefetching, memory parallelism, pointer-chasing code

9 TRIPS: A polymorphous architecture for exploiting ILP, TLP, and DLP

Karthikeyan Sankaralingam, Ramadass Nagarajan, Haiming Liu, Changkyu Kim, Jaehyuk Huh, Nitya

Robert G. McDonald, Charles R. Moore

March 2004

ACM Transactions on Architecture and Code Optimization (TACO), Volume 1 Issue 1

Full text available:  [pdf\(832.30 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citation](#)

This paper describes the *polymorphous* TRIPS architecture that can be configured for different architectures. This architecture is the first in a class of post-RISC, dataflow-like instruction sets called explicit data-gr with hardware mechanisms that enable the processing cores and the on-chip memory system to be instruction, data, or thread-level parallelism. To adapt ...


Keywords: Computer architecture, configurable computing, scalable and high-performance computing

10 The white dwarf: a high-performance application-specific processor

A. Wolfe, M. Breternitz, C. Stephens, A. L. Ting, D. B. Kirk, R. P. Bianchini, J. P. Shen

May 1988

ACM SIGARCH Computer Architecture News , Proceedings of the 15th Annual Conference on Computer Architecture, Volume 16 Issue 2

Full text available:  pdf(1.40 MB)


Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This paper presents the design and implementation of a high-performance special-purpose processor for element analysis algorithms. The White Dwarf CPU contains two Am29325 32-bit floating-point processors and employs a wide-instruction word architecture in which the application algorithm is directly implemented. It is compatible and interfaces with a SUN 31160 host. The system ...

11 Distributed operating systems

Andrew S. Tanenbaum, Robbert Van Renesse

December 1985 **ACM Computing Surveys (CSUR)**, Volume 17 Issue 4

Full text available:  pdf(5.49 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Distributed operating systems have many aspects in common with centralized ones, but they also have many differences. This paper gives an introduction to distributed operating systems, and especially to current university research about distributed operating systems and how it is distinguished from a computer network, various key developments of current research projects are examined in some detail ...

12 Increasing the instruction fetch rate via multiple branch prediction and a branch address cache

Tse-Yu Yeh, Deborah T. Marr, Yale N. Patt

August 1993 **Proceedings of the 7th international conference on Supercomputing**

Full text available:  pdf(1.13 MB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#), [review](#)

13 A parallel embedded-processor architecture for ATM reassembly

Richard F. Hobson, P. S. Wong

February 1999 **IEEE/ACM Transactions on Networking (TON)**, Volume 7 Issue 1

Full text available:  pdf(331.21 KB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

Keywords: ATM, embedded systems, medium access control, segmentation and reassembly

14 Embedded applications: AES and the cryptonite crypto processor

Dino Oliva, Rainer Buchty, Nevin Heintze

October 2003 **Proceedings of the 2003 international conference on Compilers, architecture and cryptography**

Full text available:  pdf(346.09 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

CRYPTONITE is a programmable processor tailored to the needs of crypto algorithms. The design of the processor is based on an application analysis in which standard crypto algorithms (AES, DES, MD5, SHA-1, etc) were distilled into a common methodology and use AES as a central example. Starting with a functional description of AES, we present an efficient hardware implementation of AES, and present several novel optimizations (which ...

Keywords: AES, architecture, cryptography, high-bandwidth, high-speed, processor, round key generation

15 Performance evaluation and improvement of a dynamically microprogrammable computer with

Shinji Tomita, Kiyoshi Shibayama, Toshiaki Kitamura, Hiroshi Hagiwara

November 1980 **Proceedings of the 13th annual workshop on Microprogramming**

A new microprogrammable computer with low-level parallelism was built and has been utilized as research-oriented applications such as real-time processings on static/dynamic images, pictures at virtual machines including high (intermediate) level language machines. The design goal of a research is a high degree of processing power and system flexibility ...

16 The Vector-Thread Architecture

March 2004 **ACM SIGARCH Computer Architecture News , Proceedings of the 31st annual ACM SIGARCH computer architecture conference**, Volume 32 Issue 2

Full text available:  pdf(317.13 KB)Additional Information: [full citation](#), [abstract](#)

The vector-thread (VT) architectural paradigm unifies the vector and multithreaded compute mode with a control processor and a vector of virtual processors (VPs). The control processor can use vector instructions to direct all the VPs or each VP can use thread-fetches to direct its own control flow. A seamless intermixing allows a VT architecture to flexibly and compactly encode application ...

17 Response Time Analysis of Multiprocessor Computers for Database Support

Roger K. Shultz, Roy J. Zingg

March 1984 **ACM Transactions on Database Systems (TODS)**, Volume 9 Issue 1

Full text available:  pdf(2.27 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Comparison of three multiprocessor computer architectures for database support is made possible by these expressions. These expressions are derived by parameterizing algorithms performed by each machine to execute queries. These expressions represent properties of the database and components of the machines. Studies of particular parameters of conventional machine technology, for low selectivity, high duplicate occurrence, ...

18 Retrieval operations and data representations in a context-addressed disc system

Stanley Y. W. Su, George P. Copeland, G. Jack Lipovski

November 1973 **Proceedings of the 1973 meeting on Programming languages and information systems**

Full text available:  pdf(1.15 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This paper attempts to demonstrate that simple expansion of the processing capabilities of fixed disk mappings from high-level retrieval language to machine language and from user oriented data representation to oriented data representation (storage structure) which are found necessary in conventional von Neumann architecture. The built in the disc read and write heads for each disc track allow information ...

19 System-level power optimization: techniques and tools

Luca Benini, Giovanni de Micheli

April 2000 **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Volume 5 Issue 2

Full text available:  pdf(385.22 KB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This tutorial surveys design methods for energy-efficient system-level design. We consider electrostatic and dynamic power in software layers. We consider the three major constituents of hardware that consume energy, namely, logic, memory, and interconnect, and we review methods of reducing their energy consumption. We also study models for analyzing energy consumption for energy-efficient software design and compilation. This survey ...

20 Exploiting choice: instruction fetch and issue on an implementable simultaneous multithreaded processor

Dean M. Tullsen, Susan J. Eggers, Joel S. Emer, Henry M. Levy, Jack L. Lo, Rebecca L. Stamm

May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual ACM SIGARCH computer architecture conference**, Volume 24 Issue 2

Full text available:  pdf(1.48 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Simultaneous multithreading is a technique that permits multiple independent threads to issue multiple instructions per clock cycle. This paper demonstrates the performance potential of simultaneous multithreading, based on a somewhat idealized model. The throughput gains from simultaneous multithreading can be achieved *without* extensive changes to hardware structures or sizes. We present an architecture for simultaneous multithreading ...